UNIVERSIDADE PRESBITERIANA MACKENZIE
Faculdade de Computação e Informática

CAROLINA MOLINARI MONTEFORTE
LEANDRO DA CRUZ CIRQUEIRA
LEVY SALLES BISPO DE OLIVEIRA
WILLIAM SILVA VEÇOSO

# Aprimorando a Experiência do Usuário: Uma Abordagem de Aprendizado de Máquina em Sistemas de Recomendação de Filmes

São Paulo, 2024

## LISTA DE ILUSTRAÇÕES

**LISTA DE TABELAS**

**SUMÁRIO**

# 1 INTRODUÇÃO

## 1.1 Contexto

Na indústria cinematográfica, algoritmos de recomendação de filmes têm ganhado importância com a popularidade das plataformas de streaming e o volume de dados gerados nesses serviços. Com a quantidade de conteúdo produzida pelos apps de streaming e a variedade de títulos, encontrar conteúdo relevante com os interesses pessoais de cada um tornou-se um desafio para os usuários.

## 1.2. Motivação

Diante do contexto apresentado, entender o que torna um filme atraente para diferentes públicos é essencial. A motivação para o desenvolvimento de um sistema de recomendação de filmes surge da necessidade de proporcionar uma experiência personalizada e acessível para diferentes públicos, promovendo a inclusão digital e cultural, ao mesmo tempo em que facilita o acesso a conteúdos diversos e relevantes, contribuindo para a democratização do entretenimento e o engajamento da comunidade com tecnologias avançadas de recomendação

## 1.3 Justificativa

A criação de um sistema de recomendação de filmes justifica-se pelo impacto positivo na inclusão digital e no acesso democrático à cultura, promovendo a disseminação de conhecimento sobre tecnologias de recomendação e facilitando o acesso a conteúdos audiovisuais relevantes, beneficiando tanto usuários individuais quanto a comunidade em geral por meio de possíveis ações educativas e sociais.

Este trabalho utiliza o TMDB Movie Dataset, que inclui dados sobre o enredo, elenco, equipe técnica, orçamento e receitas de vários filmes. A base de dados foi gerada pelo TMDB API, e oferece uma oportunidade de desenvolver um sistema de recomendação de filmes que atendam aos gostos e expectativas dos usuários, justificando a criação deste sistema com a necessidade atual de otimizar a experiência de escolha e melhorar o engajamento com o conteúdo.

## 1.4 Objetivos

- Desenvolver um sistema de recomendação de filmes, para facilitar a escolha e o acesso a conteúdos audiovisuais relevantes para cada pessoa.

- Avaliar diferentes algoritmos de recomendação (colaborativos e baseados em conteúdo) para determinar o mais eficiente na recomendação de filmes.
- Identificar possíveis inconsistências nos dados, como dados nulos ou incorretos, e propor métodos de correção para garantir a precisão do sistema de recomendação.

## 2 REFERENCIAL TEÓRICO

Segundo Carlos A. Gomez-Uribe e Neil Hunt (2015), o sistema de recomendação da Netflix influência em 80% das horas assistidas no serviço de streaming. Além disso, segundo um estudo descrito pelo jornal The Wrap, em 2016, o tempo médio que o usuário leva para escolher um título da Netflix é quase o dobro do que um usuário de TV a cabo leva para escolher um canal. Esses fatos apresentados reforçam a justificativa em desenvolver tais sistemas.

Para construí-los, notamos que os sistemas de recomendação podem ser aplicados de diversas maneiras, não existindo uma solução única para ser utilizada em todas as situações. A escolha do método depende bastante do tipo de informações disponíveis. Ao falarmos de filmes, por exemplo, temos muitas informações disponíveis como título, ano, diretores, atores, gênero, avaliações de uma determinada rede de usuários e assim por diante. As avaliações podem ser explícitas (como “curtidas”, “gostei ou não gostei”, estrelas) ou implícitas (tempo assistido). As abordagens mais conhecidas são as de filtragem colaborativa e a baseada em conteúdo.

A filtragem colaborativa se baseia na premissa de que usuários com gostos similares tendem a gostar dos mesmos itens. Ela analisa as interações entre usuários e itens para identificar padrões e fazer sugestões. Porém, pode ter dificuldades quando há poucos dados sobre um novo usuário ou item.

A filtragem baseada em conteúdo utiliza as características dos itens para fazer recomendações, analisando o que o usuário já gosta e buscando itens similares. Essa técnica é útil para lidar com a falta de informações sobre novos usuários ou itens, mas pode contribuir para um “efeito de bolha”, mostrando sempre conteúdos muito similares e ignorando a evolução dos gostos dos usuários.

Como podemos observar, as duas técnicas citadas possuem vantagens e desvantagens. Porém, há uma terceira técnica que mistura as duas citadas. Segundo Burke (2002), a filtragem híbrida combina ambas as técnicas, buscando os pontos fortes de cada uma e mitigando suas limitações. Essa abordagem é considerada a mais robusta e é amplamente utilizada em sistemas de recomendação comerciais.

Ainda segundo Burke (2002), a filtragem colaborativa é provavelmente a mais familiar e a mais amplamente implementada e madura das tecnologias de filtragem. Levando em consideração essa citação, optamos por utilizar a filtragem colaborativa com o algoritmo SVD como primeira abordagem deste trabalho. Já no processo de melhoria, utilizamos um sistema híbrido com o SVD e o algoritmo KNNBasic combinados, a fim de melhorar as predições.

## 3 METODOLOGIA

Tendo em vista os objetivos propostos, será utilizado Python como linguagem de programação com o apoio dos seguintes pacotes:

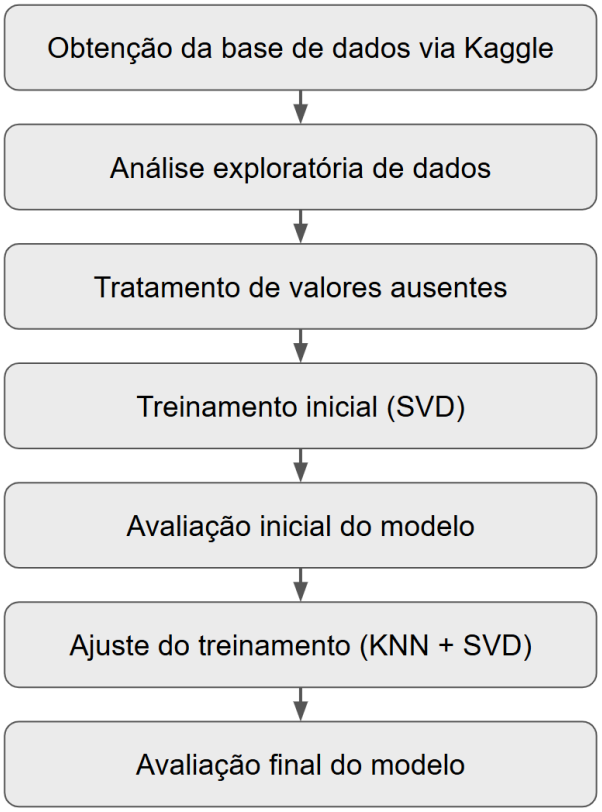
- pandas
- numpy
- matplotlib.pyplot
- seaborn
- ast
- wordcloud
- surprise

Esse projeto será armazenado em um repositório do Github, podendo ser acessado pelo link abaixo:
https://github.com/CirqueiraLeandro/CLLW-PROJETO-III

A apresentação dos resultados e das conclusões pode ser vista também no Youtube em: https://youtu.be/iW5evcX2P0Q.

O projeto seguirá o esquema geral da metodologia apresentado na figura abaixo:

Figura 1 - Metodologia utilizada no projeto




Fonte: os autores.

### 3.1 Análise exploratória de dados e tratamento de valores ausentes

Iniciamos a análise exploratória dos dados com o objetivo de entender quais são os dados disponíveis nesse projeto, descrevendo e sumarizando suas principais características (disponibilizadas na seção 4).

Primeiramente realizamos uma análise e tratamento de valores ausentes, além de remover algumas colunas que não seriam relevantes na análise (imdb_id ,original_title, adult). Em seguida trabalhamos com uma nuvem de palavras a fim de entender a prevalência de palavras chaves nos títulos e sinopses dos filmes presentes no dataset.

Também exploramos a distribuição das notas de popularidade, contagem de votos e média de votos, bem como suas medidas de centralidade.

### 3.2 Treinamento do modelo

O método selecionado para esse projeto tem como base a técnica de fatoração de matrizes pela decomposição em valores singulares, ou Singular Value Decomposition (SVD).

A SVD é uma técnica matemática que permite "desembaraçar" dados complexos em componentes mais simples. Diminuindo a quantidade de informações necessárias para representar os dados, tornando os cálculos mais eficientes.

A matriz de avaliações representa as notas que os usuários dão aos filmes, sendo decomposta em duas matrizes menores: uma de usuários e uma de filmes. Cada linha nas matrizes representa um "vetor latente", que corresponde a um conceito ou característica (por exemplo, gênero, ator favorito). Ao calcular o produto escalar entre os vetores latentes de um usuário e um filme, podemos prever a nota que o usuário daria para aquele filme.

A SVD permite que os sistemas de recomendação entendam as preferências dos usuários identificando os gêneros, atores ou outros fatores que os usuários mais apreciam, recomenda filmes relevantes que se encaixam no perfil de cada usuário, mesmo que eles não tenham visto filmes semelhantes antes, e pode lidar com dados esparsos, podendo utilizar matrizes com muitas entradas faltantes (quando um usuário não avaliou um filme), o que é comum em sistemas de recomendação.

Optamos por utilizar a biblioteca Surprise, já que ela é uma biblioteca do Python dedicada a algoritmos de predição e recomendação. Dessa maneira, a função matemática SVD já está disponível como função dentro da Surprise e avaliado por meio de validação cruzada com 5 divisões (folds). A validação cruzada foi realizada dividindo os dados em 5 partes. A cada iteração, o modelo foi treinado em 4 partes e testado na parte restante, de forma que cada subconjunto fosse usado tanto para treino quanto para teste ao longo do processo. Esse método fornece uma medida mais confiável da performance do modelo, uma vez que utiliza a totalidade dos dados.

### 3.3 Melhoria do modelo e reavaliação de desempenho

Após análise dos resultados preliminares, optamos por buscar formas de otimizar o modelo inicial. Segundo Pedro Chamberlain Matos (2021), "para fazermos uso do melhor poder algorítmico dos sistemas apresentados neste resumo até agora, nós podemos confeccionar sistemas de recomendação híbridos que juntam dois algoritmos de tipos diferentes" (p. 22). Desta forma, traçamos como rota tornar o modelo híbrido, combinando dois algoritmos: SVD + KNN, mais especificamente sua implementação básica, conhecida como KNNBasic.

Aplicamos o GridSearchCV para encontrar os melhores parâmetros entre uma série de combinações, otimizando as métricas de erro (RMSE e MAE). Com os melhores parâmetros, o modelo SVD foi re-treinado e testado.

Em seguida aplicamos um modelo híbrido, combinando o SVD otimizado e o modelo KNN. Em cada iteração do K-Fold, ambos os modelos faziam previsões, e suas saídas eram usadas como entradas para um modelo meta (Regressão Linear).

O KNN é um algoritmo de aprendizado de máquina supervisionado que classifica novos pontos de dados com base em seus 'K' vizinhos mais próximos em um conjunto de treinamento. Em termos simples, o algoritmo busca os 'K' elementos de dados mais similares a um novo dado e, em seguida, atribui a este novo dado a classe que é mais comum entre seus 'K' vizinhos. O algoritmo apresenta vantagens já que não requer treinamento intensivo e é adaptável a diferentes métricas de distância.

# 4 RESULTADOS

## 4.1 Metadados

**Objetivo:** Entender quais são os dados disponíveis nesse projeto, descrevendo e sumarizando suas principais características.

**Nome da tabela:** movies_metadata
**Descrição:** Base de dados de 45000 filmes contidos no dataset Full MovieLens
**Origem:** https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset
**Tipo de arquivo:** csv
**Sensibilidade:** Não há dados sensíveis
**Validade:** Não há validade
**Proprietário do dado:** Público
**Restrições de uso:** Não há restrições de uso
**Número de colunas:** 24
**Número de linhas :** 45466

Tabela 1 - dicionário de metadados

| Dicionário de metadados - movies_metadata | | | | | |
|---|---|---|---|---|---|
| | **Variável** | **Tipo** | **Exemplo** | **%Nulos** | **Origem** |
| **0** | adult | object | False | 0.00 | movies_metadata |
| **1** | belongs_to_collection | object | {'id': 10194, 'name': 'Toy Story Collection', ... | 90.12 | movies_metadata |
| **2** | budget | object | 30000000 | 0.00 | movies_metadata |
| **3** | genres | object | [{'id': 16, 'name': 'Animation'}, {'id': 35, '... | 0.00 | movies_metadata |
| **4** | homepage | object | http://toystory.disney.com/toy-story | 82.88 | movies_metadata |
| **5** | id | object | 862 | 0.00 | movies_metadata |
| **6** | imdb_id | object | tt0114709 | 0.04 | movies_metadata |
| **7** | original_language | object | en | 0.02 | movies_metadata |
| **8** | original_title | object | Toy Story | 0.00 | movies_metadata |
| **9** | overview | object | Led by Woody, Andy's toys live happily in his ... | 2.10 | movies_metadata |
| **10** | popularity | object | 21.946943 | 0.01 | movies_metadata |
| **11** | poster_path | object | /rhIRbceoE9lR4veEXuwCC2wARtG.jpg | 0.85 | movies_metadata |

| | | | | | |
|---|---|---|---|---|---|
| **12** | production_companies | object | [{'name': 'Pixar Animation Studios', 'id': 3}] | 0.01 | movies_metadata |
| **13** | production_countries | object | [{'iso_3166_1': 'US', 'name': 'United States o... | 0.01 | movies_metadata |
| **14** | release_date | object | 1995-10-30 | 0.19 | movies_metadata |
| **15** | revenue | float64 | 373554033.0 | 0.01 | movies_metadata |
| **16** | runtime | float64 | 81.0 | 0.58 | movies_metadata |
| **17** | spoken_languages | object | [{'iso_639_1': 'en', 'name': 'English'}] | 0.01 | movies_metadata |
| **18** | status | object | Released | 0.19 | movies_metadata |
| **19** | tagline | object | NaN | 55.10 | movies_metadata |
| **20** | title | object | Toy Story | 0.01 | movies_metadata |
| **21** | video | object | False | 0.01 | movies_metadata |
| **22** | vote_average | float64 | 7.7 | 0.01 | movies_metadata |
| **23** | vote_count | float64 | 5415.0 | 0.01 | movies_metadata |

Fonte: os autores.

**Resumo estatístico:** apresentação de medidas de tendência e dispersão

Imagem 1 - Resumo estatístico

| | count | mean | std | min | 25% | 50% | 75% | \ |
|---|---|---|---|---|---|---|---|---|
| revenue | 45460.0 | 1.120935e+07 | 6.433225e+07 | 0.0 | 0.0 | 0.0 | 0.0 | |
| runtime | 45203.0 | 9.412820e+01 | 3.840781e+01 | 0.0 | 85.0 | 95.0 | 107.0 | |
| vote_average | 45460.0 | 5.618207e+00 | 1.924216e+00 | 0.0 | 5.0 | 6.0 | 6.8 | |
| vote_count | 45460.0 | 1.098973e+02 | 4.913104e+02 | 0.0 | 3.0 | 10.0 | 34.0 | |

| | max |
|---|---|
| revenue | 2.787965e+09 |
| runtime | 1.256000e+03 |
| vote_average | 1.000000e+01 |
| vote_count | 1.407500e+04 |

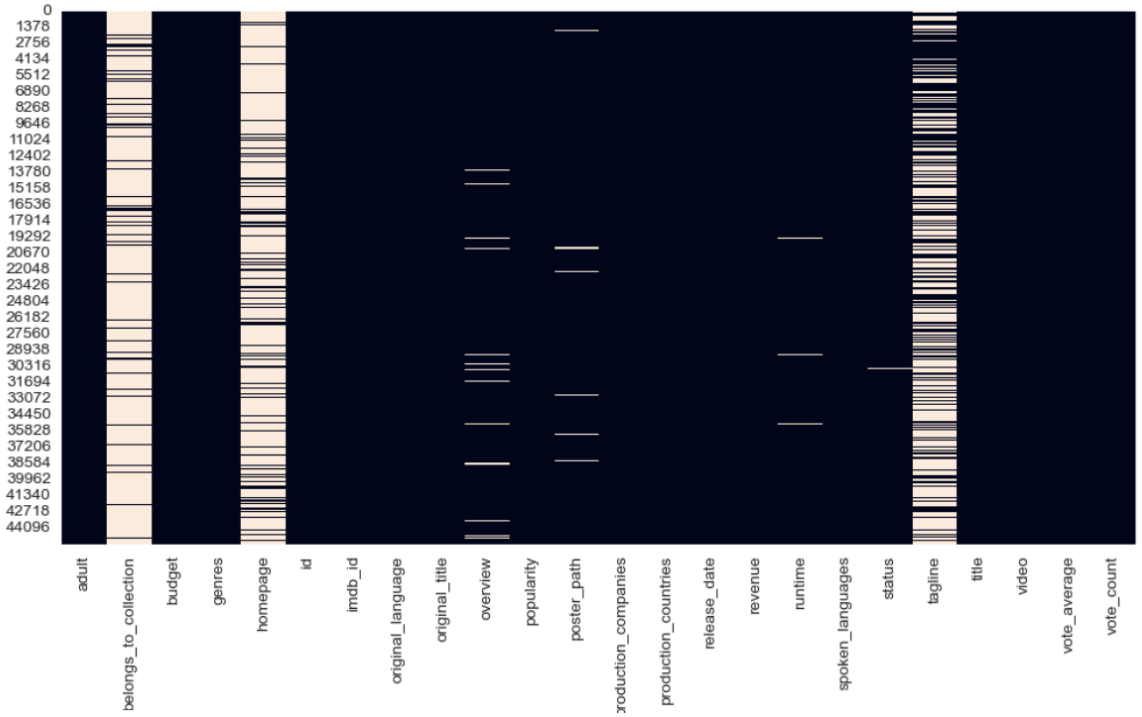
Fonte: Os autores.

### 4.2 Análise e tratamento de valores ausentes

Como pode ser visto no Gráfico 1, temos uma quantidade relevante de valores ausentes, trataremos os valores iguais a 0 para NaN nas colunas ‘revenue’ e ‘budget’.

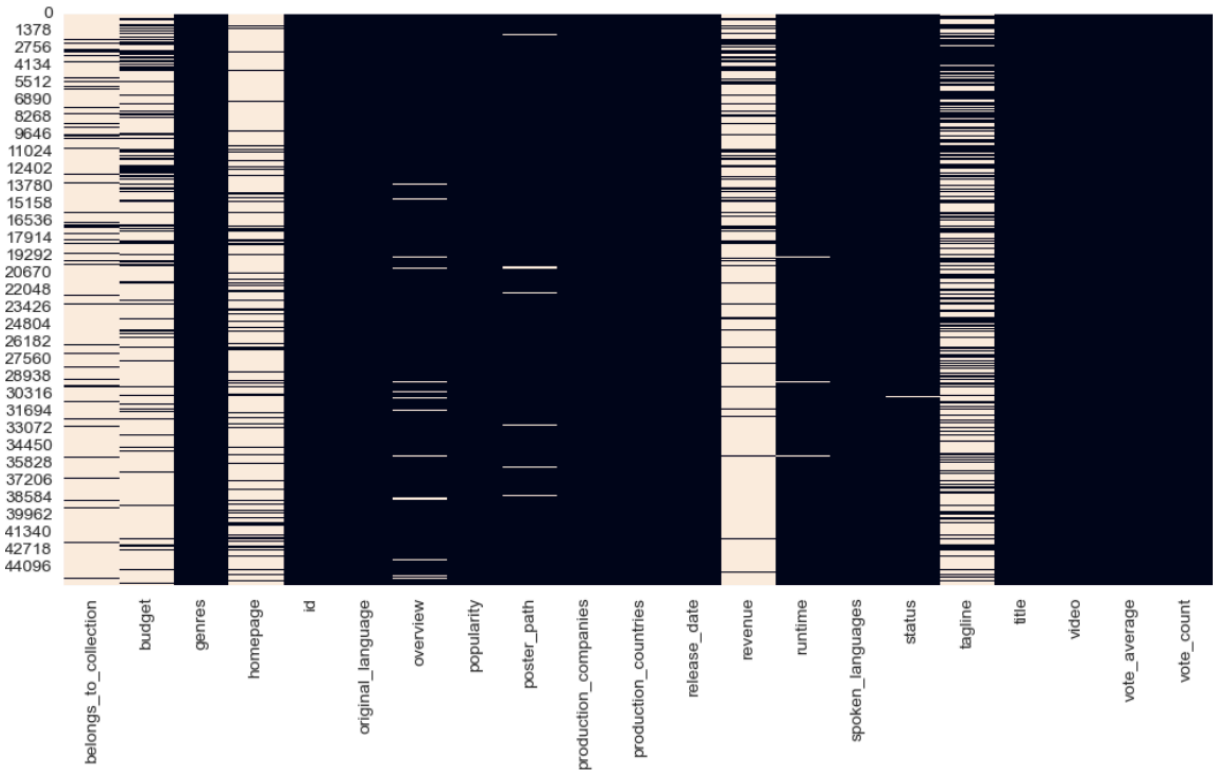
Como existem algumas colunas que não terão uso na análise elas serão removidas nessa etapa.

Gráfico 1 - Valores ausentes antes do tratamento




Fonte: os autores.

Gráfico 2 - Valores ausentes após o tratamento




Fonte: os autores.

### 4.3 Análise exploratória

#### 4.3.1 - Nuvem de palavras

Utilizamos as nuvens de palavras para entender a distribuição e frequência de palavras nos dados.

Nos títulos de filmes a palavra ‘Love’ é a mais utilizada, seguida por ‘Girl’, ‘Day’ e ‘Man’, refletindo a grande presença de romances no cinema.

Figura 2 - Nuvem de palavras nos títulos de filmes


Fonte: os autores.

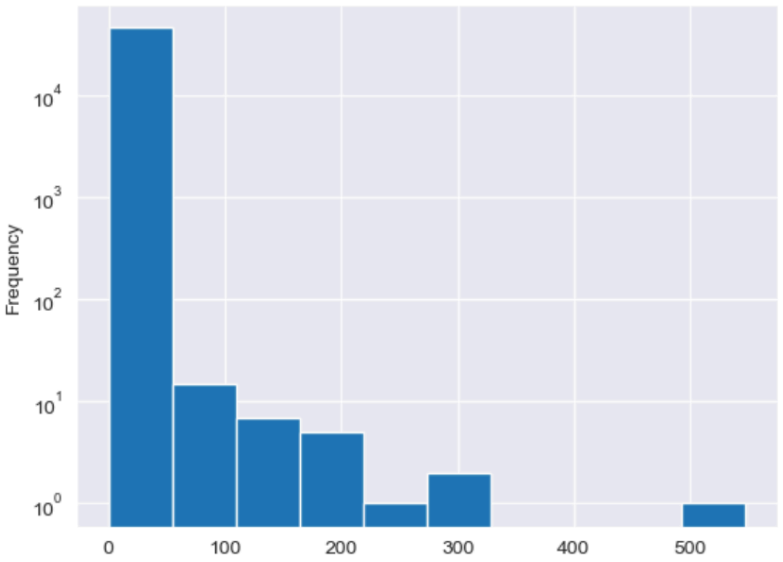
Já nas sinopses dos filmes, a palavra ‘Life’ é a mais usada, seguida de ‘One’, ‘Find’, ‘Love’, ‘Man’ e ‘Girl’. Essa sequência nos ajuda a confirmar a influência do gênero de romance no cinema.

Figura 3 - Nuvem de palavras na sinopse de filmes


Fonte: os autores.

### 4.3.2 - Popularidade

Gráfico 3 - Distribuição de frequência da pontuação de popularidade




Fonte: os autores.

A pontuação de popularidade é uma medida altamente desproporcional, com uma média de apenas 2,9, mas podendo atingir valores máximos de até 547, o que representa quase 1800% acima da média.
Contudo, como mostrado na distribuição, a grande maioria dos filmes tem uma pontuação de popularidade abaixo de 10.

### 4.3.3 - Contagem de votos

Tabela 2 - 10 filmes mais votados

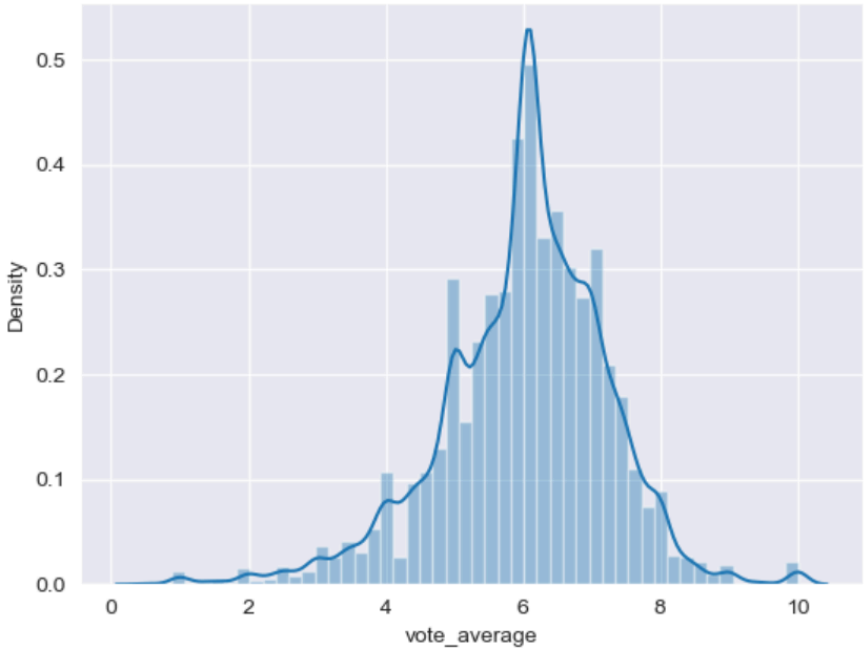
| | title | vote_count |
|---|---|---|
| **15480** | Inception | 14075.0 |
| **12481** | The Dark Knight | 12269.0 |
| **14551** | Avatar | 12114.0 |
| **17818** | The Avengers | 12000.0 |
| **26564** | Deadpool | 11444.0 |
| **22879** | Interstellar | 11187.0 |
| **20051** | Django Unchained | 10297.0 |
| **23753** | Guardians of the Galaxy | 10014.0 |
| **2843** | Fight Club | 9678.0 |
| **18244** | The Hunger Games | 9634.0 |

Fonte: os autores.

Assim como as pontuações de popularidade, a distribuição de votos é extremamente desigual, com a mediana em apenas 10 votos. O filme mais votado recebeu 14.075 votos. Assim, os votos do TMDB não são tão representativos quanto os do IMDB.
Mesmo assim, vamos conferir os filmes mais votados no site. "A Origem" e "Batman: O Cavaleiro das Trevas", dois grandes sucessos de crítica e bilheteria dirigidos por Christopher Nolan, lideram a lista.

### 4.3.4 - Média de votos

Gráfico 4 - Densidade de média de votos




Fonte: os autores.

Os usuários do TMDB avaliam os filmes de forma rigorosa, com uma média de 5,6 em 10 e metade das obras recebendo notas de 6 ou menos.

Ao analisar os filmes mais aclamados no TMDB, considerando apenas aqueles com mais de 2000 votos, "Um Sonho de Liberdade" e "O Poderoso Chefão" se destacam como os melhores. Eles também lideram a lista do IMDB, com notas acima de 9 no IMDB e 8,5 no TMDB.

Tabela 3 - 10 maiores média de votos

| | title | vote_average | vote_count |
|---|---|---|---|
| **314** | The Shawshank Redemption | 8.5 | 8358.0 |
| **834** | The Godfather | 8.5 | 6024.0 |
| **2211** | Life Is Beautiful | 8.3 | 3643.0 |
| **5481** | Spirited Away | 8.3 | 3968.0 |
| **1152** | One Flew Over the Cuckoo's Nest | 8.3 | 3001.0 |
| **1176** | Psycho | 8.3 | 2405.0 |
| **2843** | Fight Club | 8.3 | 9678.0 |
| **1178** | The Godfather: Part II | 8.3 | 3418.0 |
| **12481** | The Dark Knight | 8.3 | 12269.0 |
| **292** | Pulp Fiction | 8.3 | 8670.0 |

Fonte: os autores.

### 4.5 Avaliação do modelo

Uma vez treinado o modelo, as métricas de avaliação a serem utilizadas serão:

- RMSE (Root Mean Squared Error): Calcula a raiz quadrada da média dos erros quadráticos entre os valores preditos e os valores reais. É sensível a outliers e penaliza grandes erros.
- MAE (Mean Absolute Error): Calcula a média dos valores absolutos dos erros. É menos sensível a outliers do que o RMSE.

Os resultados obtidos em cada um dos cinco folds foram os seguintes:

- Fold 1: RMSE = 0.8873, MAE = 0.6838
- Fold 2: RMSE = 0.9085, MAE = 0.7001
- Fold 3: RMSE = 0.9055, MAE = 0.6960
- Fold 4: RMSE = 0.8934, MAE = 0.6885
- Fold 5: RMSE = 0.8921, MAE = 0.6867

Ao final da validação cruzada, as médias das métricas foram calculadas:

- Média do RMSE: 0.8921
- Média do MAE: 0.6867

A média do RMSE de aproximadamente 0.8921 indica que, em média, as previsões do modelo diferem dos valores reais em cerca de 0.9 em uma escala de avaliação que vai de 1 a 5. Esse valor sugere que o modelo é capaz de fazer recomendações com um nível de precisão satisfatório.

Além disso, o MAE médio de 0.6867 reforça essa análise, mostrando que o erro absoluto médio das previsões é de aproximadamente 0.69. Em termos práticos, isso significa que as previsões do modelo, em média, estão a menos de 1 ponto de distância das avaliações reais fornecidas pelos usuários.

### 4.5.1 Otimização de Hiperparâmetros com Grid Search

- Para melhorar o desempenho do SVD, aplicamos o GridSearchCV para encontrar os melhores parâmetros entre uma série de combinações, otimizando as métricas de erro (RMSE e MAE). Os melhores parâmetros encontrados foram:
    - n_factors: 100
    - n_epochs: 100
    - lr_all: 0.005
    - reg_all: 0.1

### 4.5.2 Modelo SVD Otimizado com GridSearchCV

- Com os melhores parâmetros, o modelo SVD foi re-treinado e testado. Na validação cruzada, os resultados melhoraram, obtendo-se:
    - **RMSE Médio**: 0.8740
    - **MAE Médio**: 0.6706
- A redução nas métricas RMSE e MAE demonstra a eficácia do ajuste fino, com uma melhora aproximada de 2% em RMSE e 2,4% em MAE em relação ao modelo inicial.

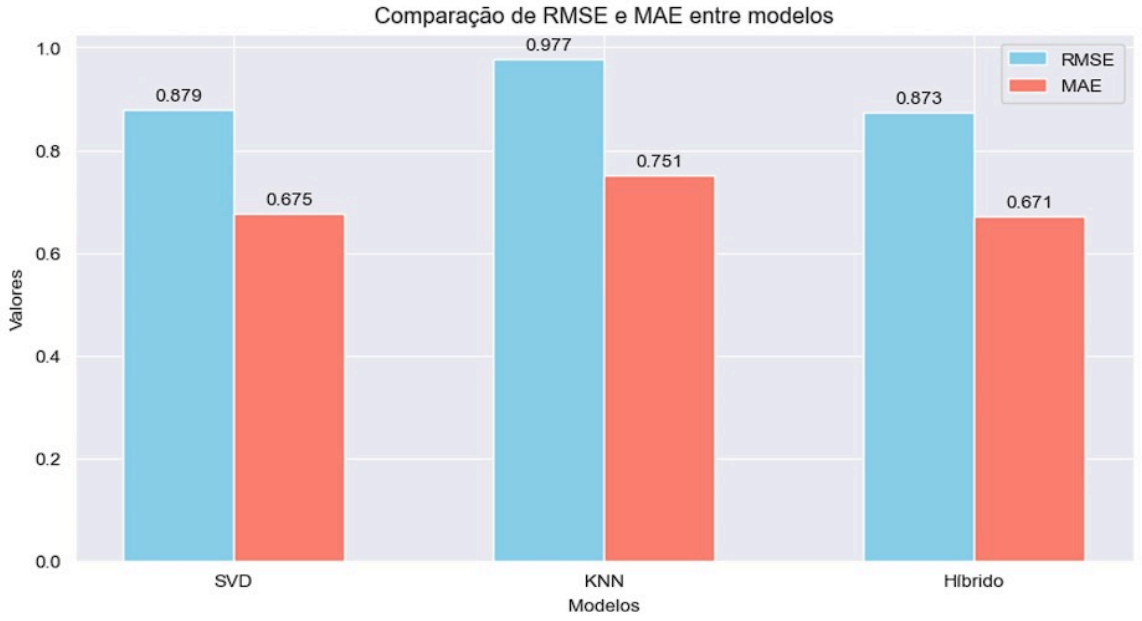
### 4.5.3 Modelo Híbrido

- Para melhorar ainda mais o modelo, aplicamos um modelo híbrido, combinando o SVD otimizado e o modelo KNN. Em cada iteração do K-Fold, ambos os modelos faziam previsões, e suas saídas eram usadas como entradas para um modelo meta (Regressão Linear).
- O resultado avaliado obteve as seguintes métricas:
    - **RMSE Médio do Stacking**: 0.8723
    - **MAE Médio do Stacking**: 0.6702
- A combinação dos modelos SVD e KNN contribuiu para uma ligeira melhora adicional, especialmente no RMSE, indicando que a combinação das técnicas de fatoração de matriz e KNN melhora a capacidade de previsão do sistema.

## 4.6 Análise dos Resultados

Vamos começar analisando os resultados dos modelos:

Gráfico 5 - Comparação de RMSE e MAE entre os modelos avaliados




Fonte: Os Autores

Conforme o Gráfico 5, conseguimos observar que o modelo SVD já era eficaz em relação às métricas RMSE e MAE, porém de fato o modelo Híbrido é o que apresenta os menores valores dessas métricas, o que indica que ele possui a menor variância entre sua previsão e os valores reais, sendo o mais eficaz para este projeto.

Gráfico 6 - Comparação entre Previsões e Avaliações Reais do Modelo Híbrido

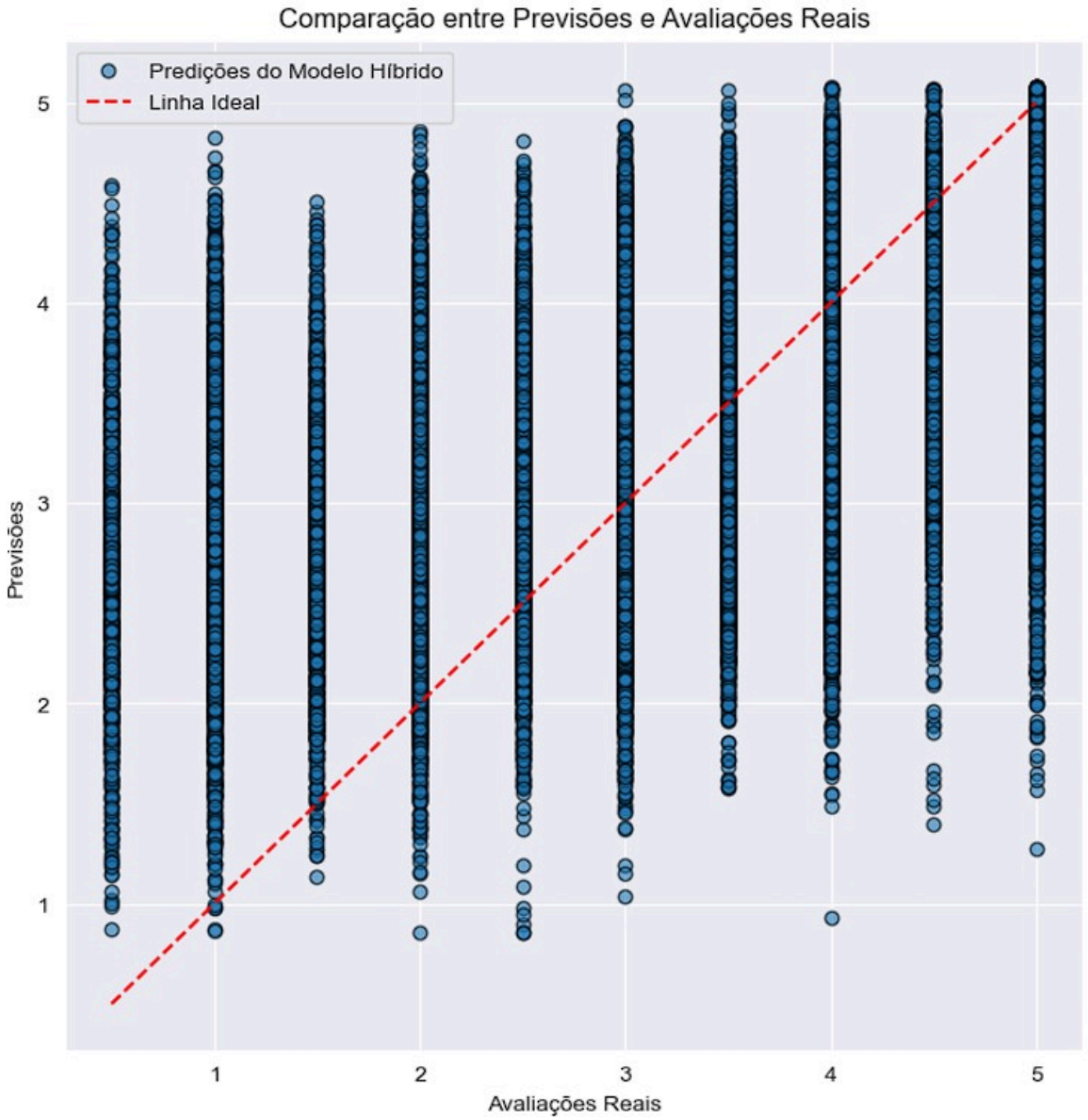
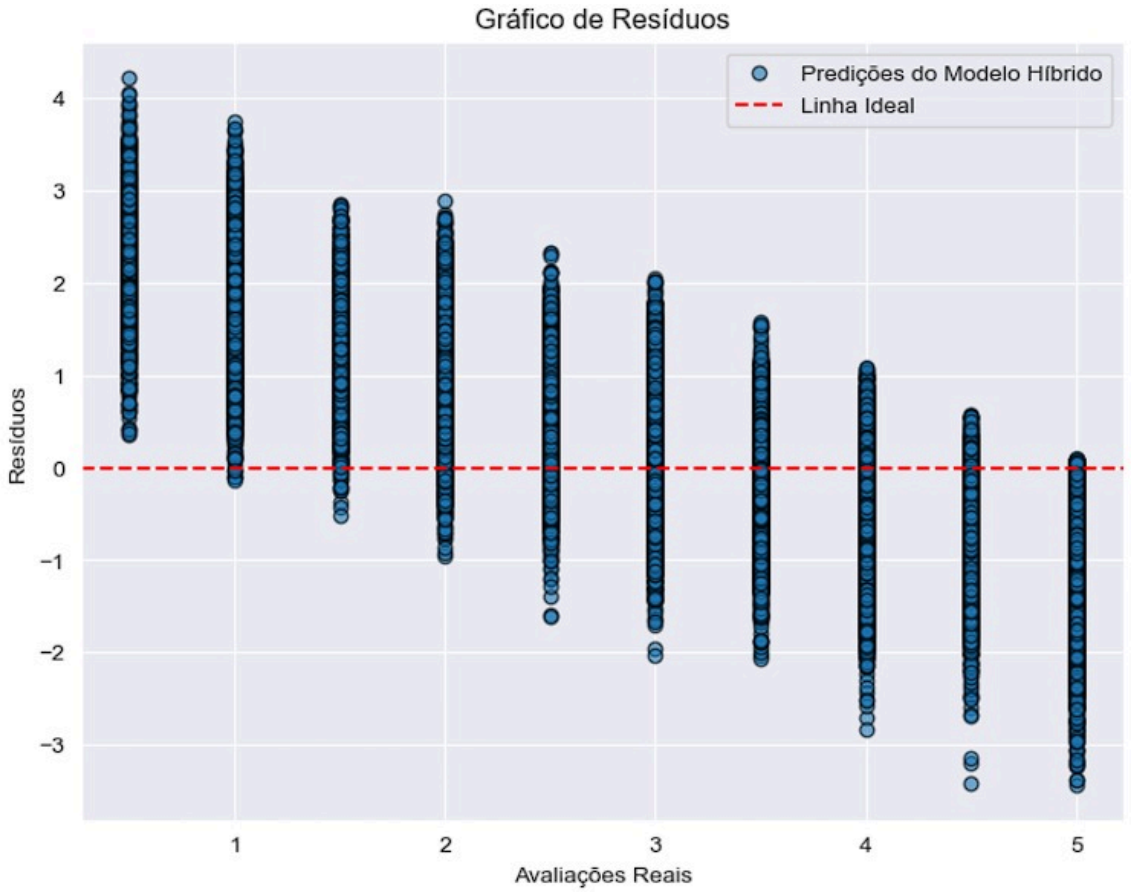
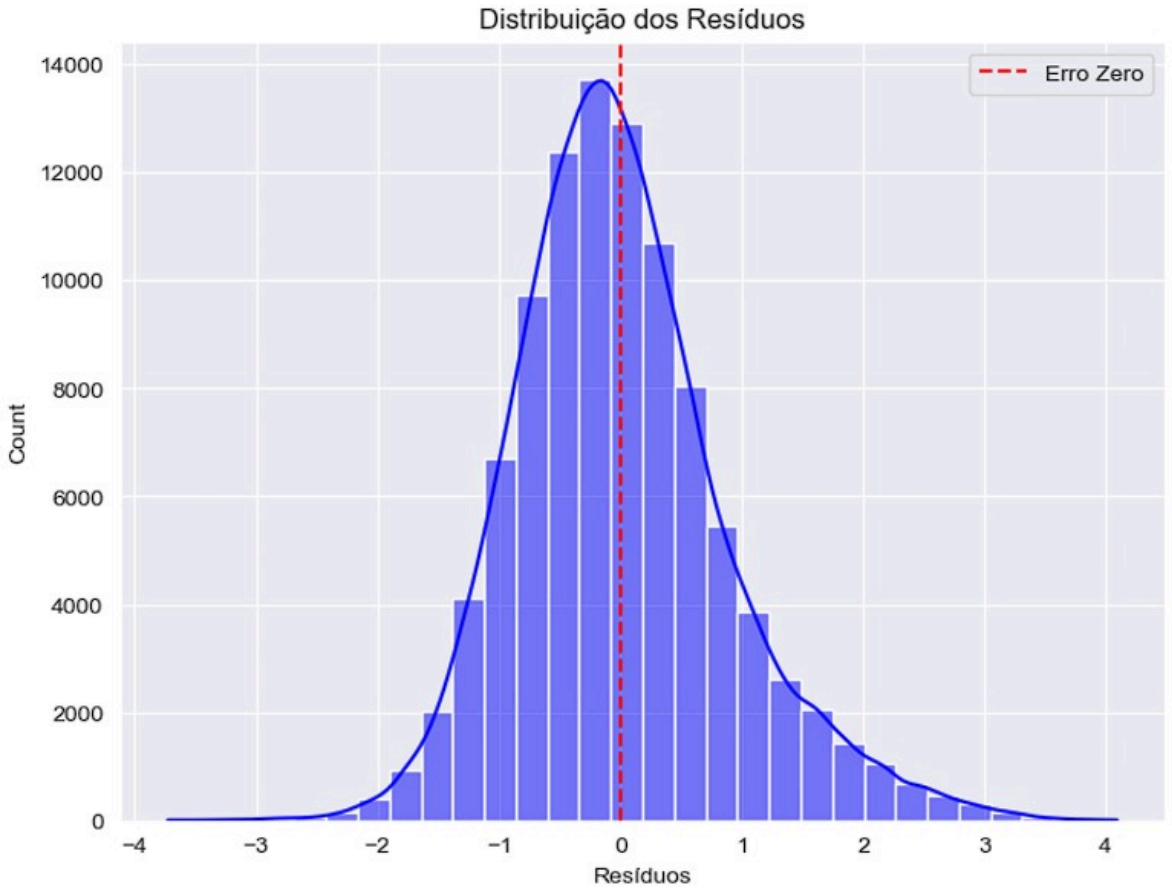



Gráfico 7 - Gráfico de Resíduos (erros entre previsões e avaliações reais)




Fontes: Os Autores

Gráfico 8 - Histograma de Distribuição dos Resíduos




Fontes: Os Autores

Analisando o Gráfico 8, fica evidente que a maior concentração dos resíduos está muito próxima de zero.

A maioria dos pontos está próxima da linha, indicando que o modelo tem boa precisão geral.

Embora existam algumas dispersões (especialmente em avaliações muito baixas ou muito altas), o modelo apresenta previsões consistentes na maioria das faixas.

O algoritmo pode apresentar limitações de cold start, já que se baseia a partir da primeira avaliação do usuário na plataforma, sendo assim, se não houver dados o suficiente para fazer recomendações elas serão bastante genéricas.

## 5 CONCLUSÕES

Avaliamos diferentes algoritmos de recomendação e determinamos o modelo híbrido como o mais eficiente na recomendação de filmes.

Iniciamos o projeto utilizando apenas o algoritmo SVD e otimizamos o treinamento através do GridSearchCV e terminamos utilizando um modelo híbrido ao adicionar o KNN e uma regressão linear o que diminuiu o erro médio em até 0.2 pontos para o RMSE e 0.165 para o MAE.

| | **SVD** | **SVD otimizado (GridSearchCV)** | **Modelo híbrido (SVD + KNN)** |
|---|---|---|---|
| RMSE | 0.8921 | 0.8740 | 0.8723 |
| MAE | 0.6867 | 0.6706 | 0.6702 |

Usando apenas o SVD, temos um algoritmo que é mais eficiente para entender relações globais entre os dados mas que pode falhar ao lidar com casos de sparsidade extrema ou itens novos. Já o KNN olha mais para os seus vizinhos mais próximos e pode fazer recomendações mais personalizadas até para os dados mais esparsos, porém pode perder escalabilidade e contexto.

Ao utilizar os dois algoritmos tentamos minimizar os pontos fracos de cada um e maximizar seus pontos fortes, o que levou a uma leve melhoria no modelo, apesar de ele ainda não lidar bem com avaliações extremas (1 e 5). Além disso, o algoritmo também enfrenta uma limitação de sugestões para usuários sem avaliações presentes na base, o que levaria a um problema de cold start.

## 6 TRABALHOS FUTUROS

Para próximas iterações no projeto, tentando resolver o problema de cold start e as avaliações extremas, pode-se adicionar informações demográficas, avaliações em sites similares, ou até solicitar palavras-chave dos usuários a fim de clusterizá-los em grupos similares ao fazer recomendações em massa e também recomendar títulos populares na plataforma.

Além disso, pode-se envolver a adoção de técnicas mais avançadas como integrar redes neurais para aprender relações mais complexas entre títulos e usuários e usar aprendizado por

reforço para otimizar recomendações com base no engajamento do usuário e técnicas auto-supervisionadas, como contrastive learning, que ajudam a identificar pares semelhantes de informações a fim de refinar as sugestões.

## 7 REFERÊNCIAS

KAGGLE. The Movies Dataset. Disponível em: <https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset?resource=download&select=movies_metadata.csv>. Acesso em: 30 set. 2024.

THE MOVIE DATABASE (TMDB) API. Disponível em: <https://developer.themoviedb.org/reference/intro/getting-started>. Acesso em: 30 set. 2024.

THE MOVIE DATABASE (TMDB). Disponível em: <https://www.themoviedb.org/>. Acesso em: 30 set. 2024.

CARLOS A. GOMEZ-URIBE AND NEIL HUNT. THE NETFLIX RECOMMENDER SYSTEM: ALGORITHMS, BUSINESS VALUE, AND INNOVATION. ACM Trans. Manage. Inf. Syst. 6, 4, Artigo 13 (Dezembro de 2015), 19 páginas.

BURKE, ROBIN. HYBRID RECOMMENDER SYSTEMS: SURVEY AND EXPERIMENTS. User Modeling and User-Adapted Interaction, Vol. 12, 2002, pp. 331-370.

PANDAS. Disponível em: <https://pandas.pydata.org/\>. Acesso em: 18 set. 2024.

NUMPY. Disponível em: <https://numpy.org/\>. Acesso em: 18 set. 2024.

MATPLOTLIB. Disponível em: <https://matplotlib.org/stable/contents.html\>. Acesso em: 18 set. 2024.

SEABORN. Disponível em: <https://seaborn.pydata.org/\>. Acesso em: 18 set. 2024.

WORDCLOUD. Disponível em: <https://pypi.org/project/wordcloud/\>. Acesso em: 18 set. 2024.

SURPRISE. Disponível em: <https://surpriselib.com/>. Acesso em: 18 set. 2024.

SCIKIT-LEARN. Disponível em: <https://scikit-learn.org/stable/\>. Acesso em: 18 set. 2024.

PREVISÃO DE AVALIAÇÕES EM SISTEMAS DE RECOMENDAÇÃO PARA NICHOS DE MERCADO. Disponível em: <https://www.cos.ufrj.br/uploadfile/1365598708.pdf>. Acesso em: 22 set. 2024.

ESTUDO COMPARATIVO DE ALGORITMOS DE SISTEMAS DE RECOMENDAÇÃO DE FILMES. Disponível em: <https://www.maxwell.vrac.puc-rio.br/57546/57546.PDF> Acesso em: 22 set. 2024.

NETFLIX USERS SPEND 18 MINUTES PICKING SOMETHING TO WATCH, STUDY FINDS. Disponível em: <https://www.thewrap.com/netflix-users-browse-for-programming-twice-as-long-as-cable-viewers-study-says/>. Acesso em: 05 out. 2024.

SCIKIT-LEARN. SKLEARN.MODEL_SELECTION.GRIDSEARCHCV. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html>. Acesso em: 30 out. 2024.

SURPRISE. SURPRISE.KNN_INSPIRED. Disponível em: <https://surprise.readthedocs.io/en/stable/knn_inspired.html>.Acesso em: 30 out. 2024.

A HYBRID COLLABORATIVE FILTERING MODEL FOR IMPROVED RECOMMENDER SYSTEM PERFORMANCE. Disponível em: <https://ijece.iaescore.com/index.php/IJECE/article/download/24045/15271>. Acesso em: 30 out. 2024.